



UNINETT

Feide Technical Guide

Technical details for integrating a service into Feide

May 2015

Document History

| Version | Date | Initials | Comments |
|---------|-----------|----------|---|
| 1.0 | Nov 2009 | TG | First issue |
| 1.2 | Nov 2009 | TG | Added SLO description |
| 1.3 | Dec 2009 | TG | Testing chapter added |
| 1.4 | Apr 2010 | TG | Adjusted to norEdu* OCS v1.5, changed logout endpoint |
| 1.5 | Dec 2010 | HV | Added information about new attribute feideSchoolList |
| 1.6 | Mar 2011 | HV | Attribute chapter rewritten |
| 1.7 | Feb 2012 | HV | Contact information for Feide updated |
| 1.8 | Sept 2014 | HV | Wrong URLs for test metadata corrected |
| 1.9 | May 2015 | JPC | Support for the HTTP-POST binding, formatting and minor fixes |
| 1.10 | Mar 2016 | JPC | Stop requiring Single Logout support |

UNINETT
Abels gate 5 – Teknobyen
NO-7465 Trondheim
telephone: +47 73 55 79 00
fax: +47 73 55 79 01
e-mail: info@uninett.no
web: www.uninett.no

Table of Contents

| | |
|--|-----------|
| 1 Introduction..... | 1 |
| 2 SAML in Feide..... | 1 |
| 2.1 SAML Messages..... | 1 |
| 2.2 Bindings..... | 5 |
| 2.3 Signing..... | 6 |
| 2.4 Profile..... | 6 |
| 3 The user's Feide session..... | 8 |
| 4 Metadata..... | 10 |
| 5 User attributes..... | 11 |
| 5.1 Information model..... | 11 |
| 5.2 Attribute availability..... | 13 |
| 5.3 SAML Assertion..... | 15 |
| 6 Testing..... | 18 |
| 6.1 Testing infrastructure..... | 18 |
| 6.2 Tests..... | 18 |
| 6.3 Troubleshooting..... | 19 |
| 7 References..... | 20 |

1 Introduction

This document is a companion to the *Feide Integration Guide* [1]. It is recommended to read that document for a conceptual and architectural introduction to Feide.

In this document you will find a more thorough discussion of the technical side of becoming a Feide integrated service provider. The first part of this document discusses how Feide implements SAML, and the next part gives an overview of the Feide user attributes.

2 SAML in Feide

Security Assertion Markup Language (SAML) [2] is a protocol framework for federated identity management. SAML is an XML-based standard.

The Feide federation consist of 3 parties:

- The Feide *Identity Provider* (IdP).
- The *Service Provider* (SP).
- The Feide user.

There are many choices left to the developers when it comes to implementing SAML inside and across organizations. *Interoperable SAML 2.0 Profile* [3] is an effort to specify a “least common denominator” for SAML federated Web based Single Sign-On (SSO). There is also a profile for single logout (SLO) deployment [4]. The Feide configuration is based on these two profiles.

There are pre-made SAML handling libraries for the major web development platforms. We urge you to use one of these unless you have a compelling reason to do otherwise. To do a robust implementation of SAML protocol handling is a major undertaking. It is written more about available software and integration in *Feide Integration Guide* [1] in chapter 5.2 *Choose and deploy SAML 2.0 Service Provider software*.

2.1 SAML Messages

During user login and logout SAML messages are communicated between the SP and IdP. How the messages are transmitted between IdP and SP is explained in chapter 2.2.

2.1.1 Login

When an SP wants to authenticate a Feide user, an *authentication request* is sent from the SP to the IdP. The message is sent by redirecting the user to the IdP web page for authentication.

```

<samlp:AuthnRequest
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ① ID="_c9c029ec886798536d71de9588668f46e7d15b1869" Version="2.0"
  ② IssueInstant="2009-10-26T13:01:03Z" ForceAuthn="false" IsPassive="false"
  ③ Destination="https://idp.feide.no/ssp/saml2/idp/SSOService.php"
  ④ ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
  ⑤ AssertionConsumerServiceURL="https://ow.feide.no/ssp/saml2/sp/acs.php">
  <saml:Issuer >
    urn:mace:feide.no:services:no.feide.openwikicore
  </saml:Issuer>
  <samlp:NameIDPolicy
    Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"
    AllowCreate="true" />
</samlp:AuthnRequest>

```

Example 1: Authentication request

A typical authentication request is shown in example 1. Key elements in the message is:

1. Unique **ID** for the request. The authentication response refers to this ID.
2. **Destination** endpoint URL for this request. This information is taken from the IdP metadata that have been imported into the SAML software.
3. **ProtocolBinding** indicates how the response message is to be transported from the IdP to the SP. Bindings are briefly explained in chapter 2.2.
4. **AssertionConsumerServiceURL** is the destination URL for the authentication response message sent back from the IdP to the SP.
5. **Issuer** is the *entityID* for this service.

Upon successful authentication the browser is by HTTP-post binding, instructed to post the authentication response message to the *AssertionConsumerServiceURL* at the SP.

```

<samlp:Response
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="_a7b867d86431b76adb83afca3e99f6192957e8183d" Version="2.0"
  IssueInstant="2009-10-26T13:04:16Z"
  Destination="https://ow.feide.no/ssp/saml2/sp/acs.php"
  ① InResponseTo="_c9c029ec886798536d71de9588668f46e7d15b1869">
  ② <saml:Issuer>https://idp.feide.no</saml:Issuer>
  <samlp:Status>
  ③ <samlp:StatusCode
    Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
  </samlp:Status>
  <saml:Assertion
    CUT: Assertion part shown separately
  </saml:Assertion>
</samlp:Response>

```

Example 2: Authentication response

A typical authentication response is shown in example 2. The *Assertion* part of the message is separated out and shown in example 3.

Key elements in the message are:

1. **InResponseTo** refers to the message *ID* in the *AuthnRequest* message.
2. **Issuer** is the *entityID* for the IdP.

```

<saml:Assertion
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  ID="pfx6a5dfea5-0003-da58-55b0-fd94ff968018" Version="2.0"
  IssueInstant="2009-10-26T13:04:16Z">
  <saml:Issuer>https://idp.feide.no</saml:Issuer>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    CUT: Lot of signature markup
  </ds:Signature>
  <saml:Subject>
    <saml:NameID
      SPNameQualifier="urn:mace:feide.no:services:no.feide.openwikicore"
      Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient">
      _508ddf0c3974b7a5951f5879e0796f97be449fcfd
    </saml:NameID>
    <saml:SubjectConfirmation
      Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
      <saml:SubjectConfirmationData
        NotOnOrAfter="2009-10-26T13:09:16Z"
        Recipient="https://ow.feide.no/ssp/saml2/sp/acs.php"
        InResponseTo="_c9c029ec886798536d71de9588668f46e7d15b1869"/>
      </saml:SubjectConfirmation>
    </saml:Subject>
  <saml:Conditions
    NotBefore="2009-10-26T13:03:46Z"
    NotOnOrAfter="2009-10-26T13:09:16Z">
    <saml:AudienceRestriction>
      <saml:Audience>
        urn:mace:feide.no:services:no.feide.openwikicore
      </saml:Audience>
    </saml:AudienceRestriction>
  </saml:Conditions>
  <saml:AuthnStatement
    AuthnInstant="2009-10-26T13:04:16Z"
    SessionNotOnOrAfter="2009-10-26T21:04:16Z"
    SessionIndex="_64da5b6b8235a8f13433e1604a1e0b31c1cd1bbb7d">
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:Password
      </saml:AuthnContextClassRef>
    </saml:AuthnContext>
  </saml:AuthnStatement>
  <saml:AttributeStatement>
    CUT: Lots of attribute markup shown separately
  </saml:AttributeStatement>
</saml:Assertion>

```

Example 3: Assertion part

The value of the **StatusCode** indicate whether the authentication was successful or not.

Key elements in the assertion part are:

1. The assertion is signed to prevent tampering of the message.
2. **SessionNotOnOrAfter** indicates the lifetime of this assertion. Session timing is discussed in chapter 3.
3. The **SessionIndex** is an ID for the IdP session.
4. The value of **AuthnContextClassRef** indicates how the user was authenticated. Feide only supports password at the moment, but this may change in the future.
5. In the *Assertion* you will also find an **AttributeStatement**. The attributes transmitted to the SP is contained in the *AttributeStatement*. You find a description of the formatting of the attributes in chapter 5.3.

2.1.2 Logout

Feide supports logout from one single service at a time or Single Logout (SLO) from all the services in the Feide federation where the user is logged in.

A *LogoutRequest* message is sent to an entity to request a logout from that entity. That entity responds by sending a *LogoutResponse* message. The *LogoutResponse* indicates whether the logout was successful or not. Depending on the mode of logout the SP may send a *LogoutRequest* to the IdP or receive a *LogoutRequest* from the IdP. The SLO modes are explained in chapter 2.4.2.

```
<samlp:LogoutRequest
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="_00ec286b0a91d783747cf237b84cba133a76693b8c" Version="2.0"
  Destination="https://idp.feide.no/ssp/saml2/idp/slo.php"
  IssueInstant="2009-10-26T14:34:25Z">
  <saml:Issuer >
    urn:mace:feide.no:services:no.feide.openwikicore
  </saml:Issuer>
  ① <saml:NameID
    Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"
    SPNameQualifier="urn:mace:feide.no:services:no.feide.openwikicore">
    _508ddf0c3974b7a5951f5879e0796f97be449fcfd
  </saml:NameID>
  ② <samlp:SessionIndex>
    _64da5b6b8235a8f13433e1604a1e0b31c1cd1bbb7d
  </samlp:SessionIndex>
</samlp:LogoutRequest>
```

Example 4: Logout request

A logout request example is shown in example 4. Key elements in the message are:

1. **NameID** is an identification of the subject (the user) of the messages and has to

match the *NameID* in the *Assertion* message, here shown in example 3.

2. The ***SessionIndex*** is the *SessionIndex* from the login assertion.

```

<samlp:LogoutResponse
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="_e9c92b6a454e084428471dc99f8dcbcc7f98a31616"
  Version="2.0" IssueInstant="2009-10-26T14:34:26Z"
  Destination="https://ow.feide.no/ssp/saml2/sp/slo.php"
  ① InResponseTo="_00ec286b0a91d783747cf237b84cba133a76693b8c">
  <saml:Issuer>https://idp.feide.no</saml:Issuer>
  <samlp:Status>
  ② <samlp:StatusCode
      Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
  </samlp:Status>
</samlp:LogoutResponse>

```

Example 5: Logout response

A *LogoutResponse* example is shown in example 5. Key element in the message is:

1. The **InResponseTo** value is the *ID* from the *LogoutRequest* this is a response to.
2. The value **StatusCode** indicates whether the logout was successful or not.

2.2 Bindings

The bindings specify how the SAML messages are transported between the SP and IdP. As specified in the SAML profile [3], Feide only uses the HTTP-redirect and HTTP-post bindings. This means that all SAML messages are transported through the user's web browser, piggy-backed on the various HTTP requests and replies. This is called front-channel communication. It eliminates the hassle of establishing a direct communication path between SP and IdP.

The binding can be illustrated by the login transaction as shown in illustration 1.

1. The user tries to access a protected web page at the SP.
2. The SP SAML functionality instructs the user's web browser (by redirect) to visit the login page at the IdP. An encoded version of the *AuthnRequest* is appended to the URL.
3. Upon successful authentication, the web browser receives a page including a hidden form control. The authentication response is encoded in this form. The browser is also instructed by a script to post this data to the SAML consumer endpoint at the SP.

The same principle is used for logout, except that logout only uses redirects.

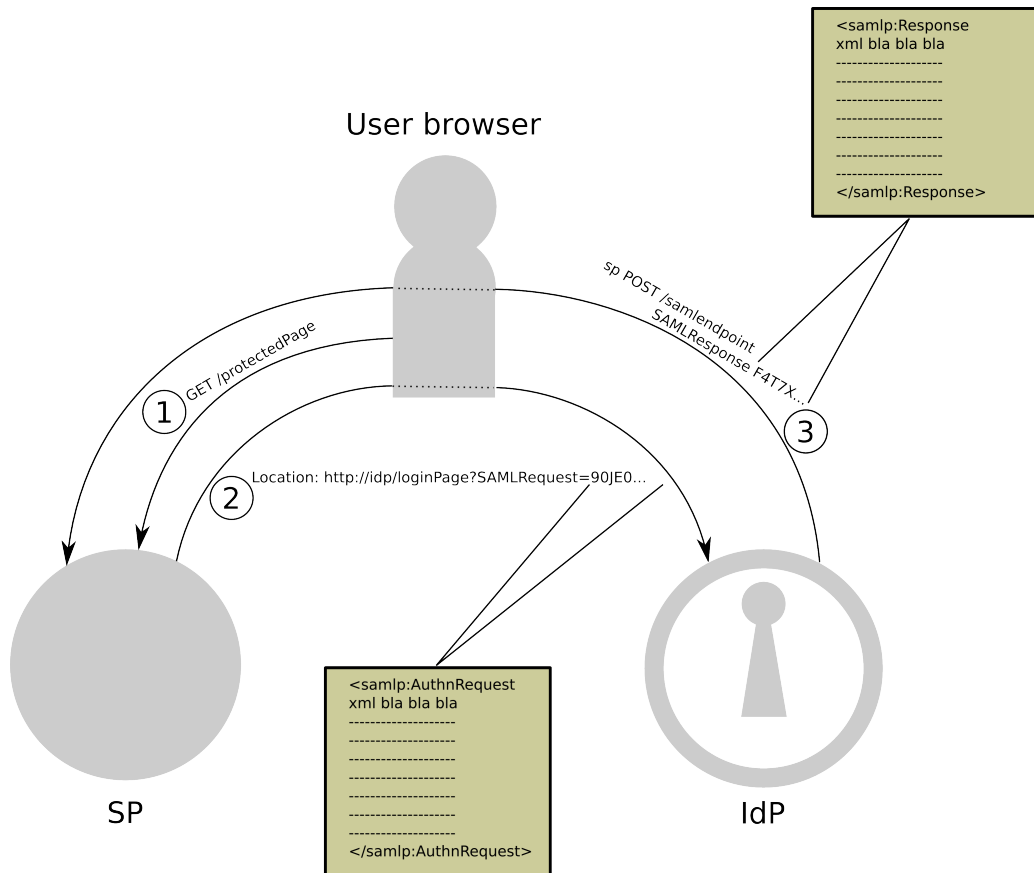


Illustration 1: SAML Message flow during login

2.3 Signing

For securing SAML communication, Feide relies on SSL/HTTPS security. This means that the SAML messages are encrypted in the communication channel between the user's browser and the service endpoint and the browser and the IdP endpoint. But the SAML messages pass through the user's web browser unencrypted. To prevent tampering of the assertion message on the user's client computer, the Feide IdP signs the assertion message in the authentication response. The Feide IdP's public certificate is embedded in the metadata for the IdP.

2.4 Profile

Profiles define specific sets of rules for using SAML for a specific task. The following chapters is a summary of the login and logout profile used in Feide.

2.4.1 Login

Feide is loosely based on the Interoperable SAML 2.0 profile specification [3]. This is a summary of the specification, adapted to its usage in Feide.

The authentication request message:

- Uses either the HTTP-redirect or the HTTP-POST binding. Please note that even though the HTTP-POST binding is allowed for sending *authentication request* messages, the use of the HTTP-redirect binding is recommended and encouraged.
- Is not signed.
- Must include AssertionConsumerURL.
- Must not include RequestedAuthnContext.
- Should, for the *NameIDPolicy*, set *AllowCreate* to *true*.
- Should, for the *NameIDPolicy*, set *nameid-format* to *transient*.

The authentication response message:

- Uses the HTTP-post binding.
- SP must support unsolicited responses. This is to support the cases where the user *bookmarks* the IdP login page.
- Includes the user attributes.
- Attributes name formats is *basic*. More information about attribute names in chapter 5.3.

SAML endpoints must use SSL/HTTPS.

2.4.2 Logout

The Feide IdP supports Single Logout (SLO). The SPs have to handle two different scenarios to properly handle SLO. Illustration 2 shows an example where one user is logged into three SPs.

When the user chooses to logout from SP1, then SP 1 initializes a logout by sending a *LogoutRequest* to the IdP. Then the user selects to logout from all the services. The IdP initializes logout on the rest of the SPs (SP 2 and SP 3) by sending a *LogoutRequest* to each of them. The SPs will perform a local logout and respond to the IdP by sending a *LogoutResponse* message back to the IdP. When the IdP has completed all the logouts it will respond to SP 1 by sending a *LogoutResponse* message to SP 1.

The user may initialize a SLO from any of the SPs. This means that any SP should be able both to send *LogoutRequest* messages and to respond to incoming *LogoutRequest* messages from the IdP.

Summary of the *Front-channel SAML 2.0 Single Logout Deployment Profile [4]*.

Recommendation for Service Provider:

- *Should be able to initialize Single Logout from the service by sending LogoutRequest message to the IdP.*

- Should be able to handle incoming *LogoutRequest* messages from the IdP. If it is able to initialize *Single Logout*, it must be able to handle incoming *LogoutResponse* messages.

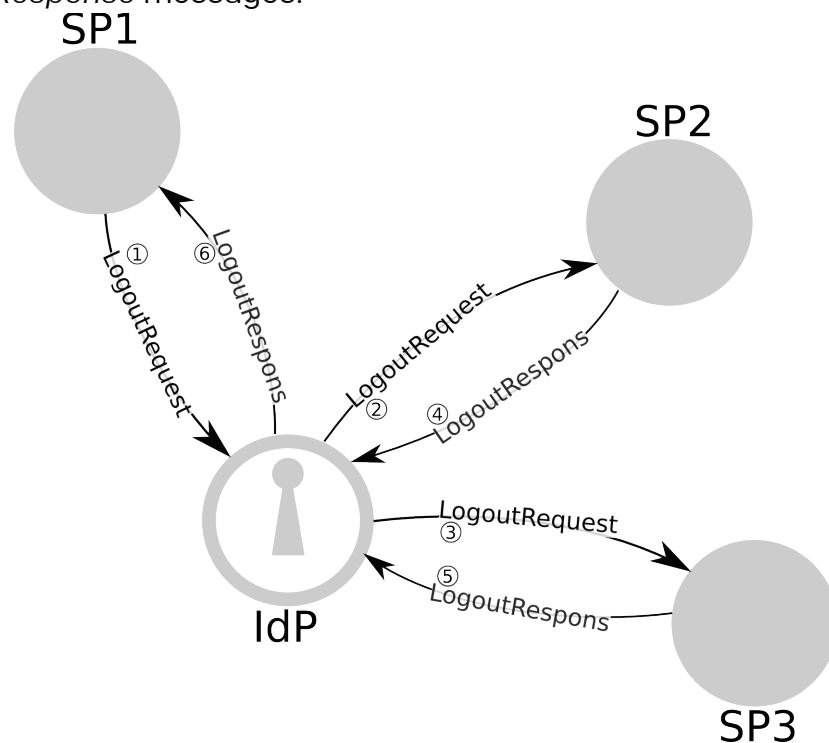


Illustration 2: SLO message flow

Feide Identity Provider:

- Handles Single Logout initiated by any of the connected Service Providers.
- Interact with the user to let the user select SLO or only logout from the IdP and the SP where logout was initialized.

The logout request message is:

- Sent by either the HTTP-redirect or HTTP-POST binding. Please note that even though the HTTP-POST binding is allowed for sending *authentication request* messages, the use of the HTTP-redirect binding is recommended and encouraged.

3 The user's Feide session

When the user does a "fresh" login on the IdP for the first time, an IdP session is created. The user's attributes are fetched from the home organization, and are cached in the session storage.

The IdP session has a lifetime of 8 hours from the username and password is entered. During this 8-hour session the user will be able to directly access services without entering the username and password again. If the user tries to access a service after the 8-hour session has expired, a new Feide login is initiated and a new 8-hour session

is created.

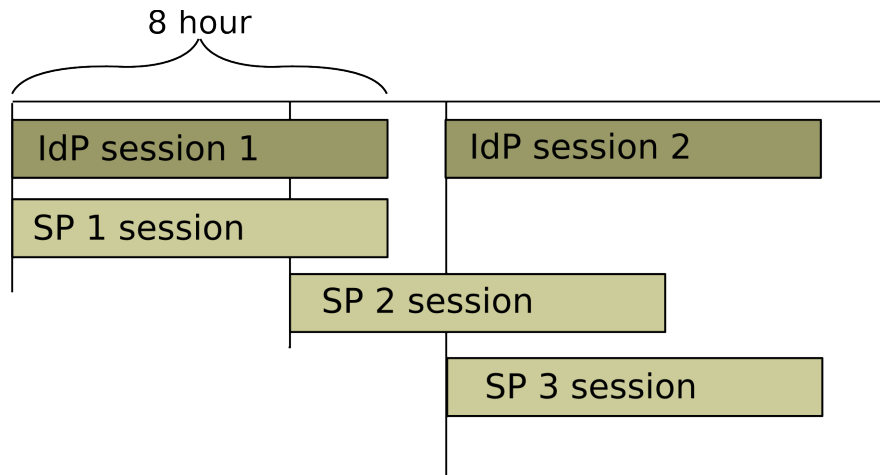


Illustration 3: Login session timing

A new session is created on each SP when the SP receives the assertion from the IdP. The SP session lifetime is also set to 8 hours.

In the assertion part in the authentication response message is an *AuthnStatement* tag (example 6). This tag contains some attributes relevant to session timings.

```
<saml:AuthnStatement
  AuthnInstant="2009-10-26T13:04:16Z"
  SessionNotOnOrAfter="2009-10-26T21:04:16Z"
  SessionIndex="_64da5b6b8235a8f13433e1604a1e0b31c1cd1bbb7d">
```

Example 6: Authentication statement

1. The **SessionNotOnOrAfter** attribute is a time stamp set to 8 hours after the authentication assertion was sent. The SP must adhere to this time limit, and initiate a new login if the user tries to access the resource after this time has expired.

The login session can be terminated by one of three ways:

1. The user selects to logout.
2. The user closes the web browser. The IdP and SP sessions cookies are automatically deleted.
3. The IdP session is automatically terminated 8 hours after the username and password was entered. The SP session is automatically terminated 8 hours after the SP received the assertion.

4 Metadata

Before your SP and the Feide IdP can operate in a federation, the entities must exchange SAML Metadata. Feide operates both a test IdP and a production IdP.

- The Feide test IdP metadata can be downloaded as XML from this link:
<https://idp-test.feide.no/simplesaml/saml2/idp/metadata.php>
- Or found formatted for web on this link:
<https://idp-test.feide.no/simplesaml/saml2/idp/metadata.php?output=xhtml>
- And the Feide production IdP metadata can be downloaded from:
<https://idp.feide.no/simplesaml/saml2/idp/metadata.php>

Metadata must be generated for your SP. This is usually done by the SAML library you are using. A SP metadata example is shown in example 7.

```
<?xml version="1.0"?>
<EntityDescriptor
  xmlns="urn:oasis:names:tc:SAML:2.0:metadata"
  ① entityID="urn:mace:feide.no:services:no.feide.openwikicore">
  <SPSSODescriptor
    protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
    ② <SingleLogoutService
      Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
      Location="https://ow.feide.no/ssp/saml2/sp/slo.php"/>
      <NameIDFormat>
        urn:oasis:names:tc:SAML:2.0:nameid-format:transient
      </NameIDFormat>
    ③ <AssertionConsumerService index="0"
      Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
      Location="https://ow.feide.no/ssp/saml2/sp/acs.php"/>
    </SPSSODescriptor>
    <ContactPerson contactType="technical">
      <GivenName>Andreas</GivenName>
      <SurName>Åkre Solberg</SurName>
      <EmailAddress>andreas@uninett.no</EmailAddress>
    </ContactPerson>
  </EntityDescriptor>
```

Example 7: Service metadata

The most important metadata entries:

1. **entityID** This is a unique ID for your service. It should either be a URN or a URL.
2. **SingleLogoutService** The *Location* attribute is the URL where logout requests and responses are handled. If this endpoint is available, it must be able to both send *LogoutRequests* and to handle incoming *LogoutRequests*.
3. **AssertionConsumerService** The *Location* attribute is the URL where authentication responses sent from the IdP will be posted.

If the *entityID* is a URL it should be a URL pointing to the metadata for the service.

An URN *entityID* should be a part of Feide's URN-namespace

"urn:mace:feide.no:services:" and consist of 3 parts separated by dot (.):

1. *no* for Norwegian service providers or the ccTLD for the country of origin of the service provider.
2. The subdomain name for the service provider.
3. A descriptive name for the service offered. The name should be unique in the service providers DNS namespace.

The *entityID* in example 7 shows a URN formatted entity id.

5 User attributes

User attributes carry different pieces of information about the user, the user's home organization and the user's school. It may be information like name, role, school affiliations etc.

When the Feide IdP has successfully authenticated the user against the user directory at the user's home organization, the Feide IdP will retrieve a set of attributes from the directory. After the Feide IdP has retrieved the attributes, it will process and filter the attributes before it passes them on according to the agreements between Feide and the SP.

5.1 Information model

There are three documents describing the Feide information model:

- The *norEdu* Object Class Specification* [5] describing how the attributes are organized. The *norEdu**-specification is based on *eduPerson*, *eduOrg* and some other LDAP schemas.
- Feide requirements for primary and secondary education.
- Feide requirements for higher education.

To understand the Feide information model, it is important to understand that the Feide federation has members from two parts of the Norwegian educational sector:

- higher education (with universities and university colleges)
- primary and secondary education

These two parts are organized different as regards to ownership and internal structures, which affects the Feide information model.

Essential to the Feide information model is that each person is associated with one top organization (the *home organization*) and possibly one or more organization units. This is shown in illustration 4.

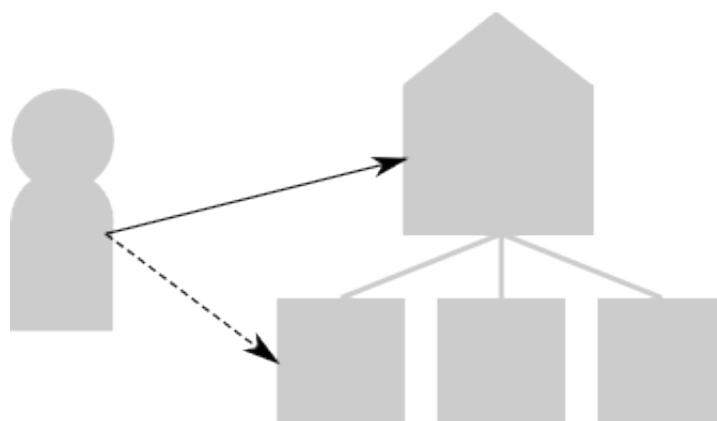


Illustration 4: All Feide users are associated with one home organization and possibly one or more organization units.

For students, employees and other persons in higher education, the home organization is the university / university college. Organization units are various faculties or departments within the top organization. There is no requirement that a person in higher education should be associated with an organization unit, and Feide puts no requirements on which attributes should be registered for organization units in higher education.

Because of different organization and owner structures in primary and secondary education, the Feide information model for primary and secondary education is different from higher education.

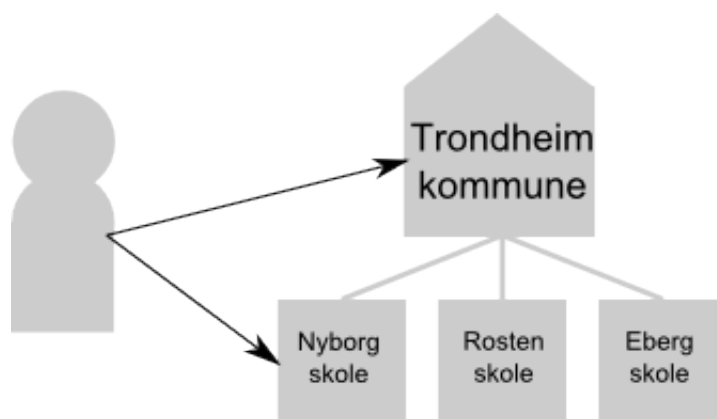


Illustration 5: A pupil must be associated with both a top organization and at least one organization unit. In this example, the top organization is Trondheim kommune (the municipality) and the organization unit is Nyborg skole (the school).

For pupils, teachers and other persons in primary and secondary education, the home organization is the owner of the school, in most cases the municipality/county. All the schools owned by the same municipality/county, are organization units under the home organization. All persons in primary and secondary education are associated with at least one organization unit (one school). Feide puts specific requirements on which attributes should be registered for organization units in primary and secondary education. An example on how this might be, is shown in illustration 5.

If the service is to do authorization based on school affiliation, it's crucial to understand how higher education and primary and secondary education is represented in Feide. See table 1 for a summary of the differences between higher education and primary and secondary education.

| | Home organization | Organization units | All users have organization units | Feide puts requirements on organization units |
|---------------------|---|---|-----------------------------------|---|
| Higher education | University, university college (ex. Universitetet i Oslo) | Various departments and faculties (ex. Faculty of Humanities) | No | No |
| Secondary education | County (ex. Rogaland fylkeskommune) | Secondary schools (ex. St. Svithun videregående skole) | Yes | Yes |
| Primary education | Municipality (ex. Trondheim kommune) | Primary schools (ex. Nyborg skole) | Yes | Yes |

Table 1: A summary of the differences between higher education and primary and secondary education.

5.2 Attribute availability

The availability of attributes is governed by the *norEdu* Object Class Specification* [5] together with the Feide requirements for primary and secondary education and higher education respectively. These documents state what information should be registered about each user and how the user directory should be organized.

The *norEdu* Object Class Specification* has been released in several versions. As a result of this, some organizations are using an older version of the specification and do not necessarily have all the attributes required by the latest specification.

A list of user attributes available for services is maintained at Feide's homepage: <http://www.feide.no/attributelist>.

5.2.1 Name

Due to the fact that not all home organization's adhere to the latest *norEdu* Object Class Specification*, there is today no single attribute that is guaranteed to hold the users name. A suggested strategy to get a name value to use in the application, is as follows:

1. Check if the user has a *displayName* and use that. If not present...
2. Check if the user has one or several *givenName* attributes; select the first one, and join with the first value from *sn* (surname) if you want to display full name. If not present...
3. Use the first value from *cn* (common name)

5.2.2 User id

When it comes to selecting a user ID we have to balance between two conflicting demands – the services' demand for a unique and persistent id and the users' demand for anonymity and privacy.

The possible user IDs should be considered in the following order:

1. *eduPersonTargetedID*: This ID gives anonymous user authentication, but still gives the service the ability to track a particular user from login session to login session. This ID is different on different services for a particular user.
2. *eduPersonPrincipalName*: This is the “Feide ID”. This is a valid and unique user ID as long as the person is a member of an educational institution.
3. *norEduPersonNIN*: This is intended as a lifelong personal identity number. In the cases where a key is needed to combine information from other governmental information sources, this ID can be used.

5.2.3 School

As described in section 5.1, higher education and primary and secondary education are organized in different ways and this affects the Feide information model. To simplify, Feide can release the attribute *feideSchoolList* to SPs that request information about the users' school. *feideSchoolList* is a multivalued attribute with the following properties:

- Contains the school's organization number, which uniquely identifies the school.
- If there is a primary school registered for the user, this school is listed first.
- For users in higher education, only the organization number of the home organization is listed.

5.2.4 Grades and subjects

For pupils in primary and secondary education it is intended to populate the

eduPersonEntitlement attribute with Grep-codes. Grep-codes can be used to express a pupil's grade, subjects, major area etc. For further information please contact Feide.

5.3 SAML Assertion

The attributes are transmitted from the IdP to the SP wrapped in SAML assertion. The SAML specification only specify an attribute name-value pair relationship. It does not specify how to map a tree like information structure into a flat name-value pair message structure. This chapter will explain how Feide organizes this information in the SAML assertion.

5.3.1 Attribute value letter case

User identity values like *eduPersonPrincipalName* is by definition case insensitive. However Feide preserve cases for attribute values and some home organizations have saved the values with both uppercase and lowercase letters. We therefore recommend to do case insensitive matching when *eduPersonPrincipalName* is used as a user identity key in the SP system.

5.3.2 Referenced objects and namespaces

All attributes carrying information about the user's home organization are prefixed by "eduPersonOrgDN", and all attributes carrying information about the user's organization units are prefixed by "eduPersonOrgUnitDN". The prefix and the attribute name is joined with colon (:). The notation is shown in example 8.

```
<saml:Attribute Name="eduPersonOrgDN:mail"  
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
```

Example 8: Attribute name

5.3.3 Multivalued attributes

The SAML assertion message snippet in example 9 shows an example where *eduPersonAffiliation* has multiple values. There is no particular ordering of the attribute values.

```
<saml:Attribute Name="eduPersonAffiliation"  
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">  
  <saml:AttributeValue xsi:type="xs:string">  
    employee  
  </saml:AttributeValue>  
  <saml:AttributeValue xsi:type="xs:string">  
    member  
  </saml:AttributeValue>  
</saml:Attribute>
```

Example 9: Multivalue attribute

5.3.4 Several orgUnits

The user may be tied to several orgUnits. The Feide IdP does some custom mapping to be able to convey this information to the service via the assertion message.

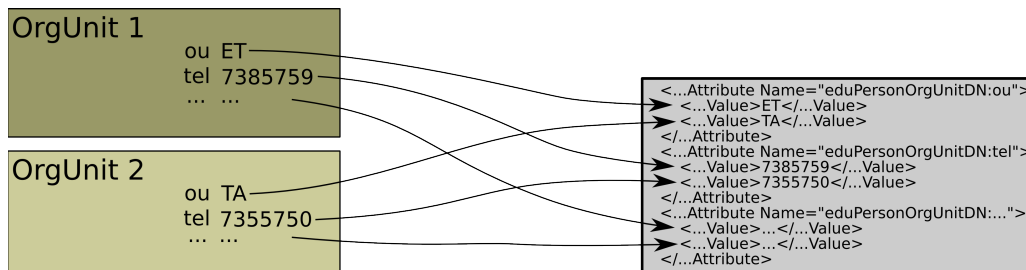


Illustration 6: Multiple org units mapping

The value for each attribute name in each orgUnit object is grouped together as a multivalued attribute in the assertion message. This mapping is illustrated in 6, and example 10 shows an example message. This mapping is done consistently, so for each attribute name in the assertion message, the first value is always from the first orgUnit object and so on. An empty value is inserted if a orgUnit object is missing this attribute.

```
<saml:Attribute Name="eduPersonOrgUnitDN:cn"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
  <saml:AttributeValue xsi:type="xs:string">
    Eksterne tjenester
  </saml:AttributeValue>
  <saml:AttributeValue xsi:type="xs:string">
    Tjenesteavdeling
  </saml:AttributeValue>
</saml:Attribute>
```

Example 10: Attribute from multiple OrgUnits

This mapping alter the semantic meaning of multivalued attributes in the assertion message for orgUnits. So in the cases where the *orgUnit* attribute is already multivalued, this has to be encoded in a different way. The multivalued attributes are joined together with the pipe (|) symbol as a separator. Example 11 shows an example where the *ou* attribute have two values.

```
<saml:Attribute Name="eduPersonOrgUnitDN:ou"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
  <saml:AttributeValue xsi:type="xs:string">
    ET|Eksterne Tjenester
  </saml:AttributeValue>
  <saml:AttributeValue xsi:type="xs:string">
    TA|Tjenestavdelingen
  </saml:AttributeValue>
</saml:Attribute>
```

Example 11: Multivalued attribute from multiple OrgUnits

6 Testing

This chapter describes which tests you should complete to verify the Feide integration of your service.

6.1 Testing infrastructure

Feide has some infrastructure to facilitate integration testing for service providers.

- Test IdP to test the SP ↔ IdP communication and metadata compatibility. The test IdP metadata is found at:

<https://idp-test.feide.no/simplesaml/saml2/idp/metadata.php?output=xhtml>

- Test SP to test single sign on (SSO) and single log out (SLO). The test SP is found at:

<https://sp-test.feide.no/>

To start testing your service, it must be integrated with the Feide test IdP. This is done by exchanging metadata with this IdP.

Feide has a test user.

| | |
|---------------------|--------|
| Organization | Feide |
| Username | test |
| Password | 098asd |

6.2 Tests

Tests no. 1 and 3 are core requirements. Tests no. 2, 4 and 5 are optional and tests your SP's ability to handle "IdP-first flow". This will occur if the user bookmarks the IdPs login page.

| no | Test | Description | Verification |
|----|--------------------|--|--|
| 1 | SSO, your SP first | While not logged in: Log in from your SP. Then log in from the Feide test SP. | When logging in to your SP, you are directed to the IdP login page and you have to enter your credentials. When logging in to the Feide test SP, you are granted access directly without entering credentials. |

| no | Test | Description | Verification |
|----|---------------------------------------|--|---|
| 2 | SLO, your SP initializes | While logged in to both your SP and Feide test SP: Start logout from your SP. Select to log out from all services when prompted at the IdP logout page. | Verify that the IdP logout page lists the Feide test SP. When logout is completed, verify that you are logged out from both services. |
| 3 | SSO, Feide test SP first | While not logged in: First log in to Feide test SP then log in to your SP. | Verify that you are granted access to your SP directly, without entering credentials, after you have logged in to the Feide test SP. |
| 4 | SLO, Feide test SP initializes | While logged in to both services: Start logout from the Feide test SP. Choose to logout from all services when prompted at the IdP logout page. | Verify that the IdP logout page lists your SP. When logout is completed, verify that you are logged out from both services. |
| 5 | Login from bookmarked IdP login page. | While not logged in: Start login from your service and bookmark the Feide login page. Restart your browser and open the bookmark. Complete the login operation. | Verify that you are granted access to your service. |

6.3 Troubleshooting

| Fault | Possible remedy |
|---|--|
| Your SP is not logged out after completed SLO in test #4. | Your SP is not configured to properly handle incoming <i>LogoutRequest</i> messages from the IdP. See [4] for further description. |

Feide IdP-test stores a cookie in your browser. The name of this cookie is *SimpleSAMLSessionID*. You can delete this cookie when you do trial and error testing, to get back to a "clean state".

For further support, you can contact Feide support at e-mail address: support@feide.no

7 References

List of referenced documents and web sites:

- [1] [Feide Integration Guide](#)
- [2] [SAML v2.0 Technical Overview \(pdf\)](#)
- [3] [Interoperable SAML 2.0 Web Browser SSO Deployment Profile](#)
- [4] [Front-channel SAML 2.0 Single Logout Deployment Profile](#)
- [5] [norEdu* Object Class Specification, PDF format](#)
- [6] [SAML 2.0 Usage in Feide](#)